Projet MicroMarket

Réalisation d’un Micro Master Data Management (MDM) au moyen d’une API

API – cURL – Import/export XLSX – export PDF

1-Présentation du projet

Le projet prévoit de gérer un référentiel de produits d’un magasin alimentaire au moyen d’une API (Application Programming Interface) REST (Representational State Transfer).

On crée des requêtes HTTP GET, POST, PUT et DELETE pour récupérer, ajouter, mettre à jour et supprimer les enregistrements de la base de données.

On peut définir des requêtes http personnalisées pour réaliser des opérations particulières.

* Définir les routes de l’application :

| **Route** | **Méthode** | **Type** | **Description** |
| --- | --- | --- | --- |
| http://127.0.0.1/api/produits | GET | JSON | Récupérer toutes les produits. |
| http://127.0.0.1/api/produits/{id} | GET | JSON | Récupérer les données d’un seul produit |
| http://127.0.0.1/api/produits | POST | JSON | Insérer un nouveau produit dans la base de données. |
| http://127.0.0.1/api/produits/{id} | PUT | JSON | Mettre à jour un produit dans la base de données. |
| http://127.0.0.1/api/produits/{id} | DELETE | JSON | Supprimer un produit de la base de données. |
| http://127.0.0.1/api/produits/{id} | DUPLICATE | JSON | Dupliquer un produit dans la base de données. |

2-Méthodologie

1. Réalisation de l’API

* Créer un répertoire ‘api/’ à la racine du projet
* Créer un répertoire ‘assets/’ à la racine du projet
* Créer une base de données appelée ‘micromarket’
* Créer une table ‘produits’ comprenant les champs suivants :

\* Un index auto-incrémenté (clé primaire), id\_product

\* Un code produit (3 ou 4 lettres en capitales ex PCC), code

\* Une description (Qq mots ex Pain de campagne aux céréales), description

\* Un prix (on a l’habitude de stocker les prix en centimes), price

\* Un identifiant de catégorie (clé étrangère), category\_id

\* Un identifiant de statut (clé étrangère), statut\_id

\* Un identifiant fournisseur (clé étrangère), supplier\_id

\* La date d’achat, purchase\_date

\* La date de péremption, expiration\_date

* Ajouter 2 ou 3 enregistrements à la main dans cette table en s’inspirant du fichier XLS joint.
* Créer une table ‘category’ comprenant les champs suivants :

\* Un index auto-incrémenté (clé primaire)

\* Un nom

* Ajouter à la main les catégories suivantes :

1. Boulangerie/Pâtisserie
2. Epicerie salée
3. Epicerie sucrée
4. Boissons
5. Fromagerie
6. Poissonnerie
7. Boucherie
8. Libre-service
9. Vente à l’étalage
10. Tête de gondole

Un produit peut avoir plusieurs catégories (Poissonnerie + Vente à l’étalage, Boucherie + Libre-service). Il faut donc créer une table de liaisons comprenant les champs suivants :

\* Un index auto-incrémenté (clé primaire)

\* Un identifiant produit

\* Un identifiant catégorie

* Créer une table ‘statut’ comprenant les champs suivants :

\* Un index auto-incrémenté (clé primaire)

\* Un nom

* Ajouter à la main les statuts suivants :

1. En cours d’approvisionnement
2. En stock
3. Epuisé
4. Retiré des rayons

* Créer une table ‘assets’ comprenant les champs suivants :

\* Un index auto-incrémenté (clé primaire)

\* Un chemin de fichier

\* Un nom de fichier

* Ajouter 2 ou 3 enregistrements à la main dans cette table.

Prendre soin de bien nommer les images (c’est important pour la suite) :

<Nom-du-produit>\_01.jpg, <Nom-du-produit>\_02.jpg, <Nom-du-produit>\_03.jpg

Un produit peut avoir plusieurs visuels (coloris, conditionnements, shoot, ambiance, etc.). Il faut donc créer une table de liaisons comprenant les champs suivants :

\* Un index auto-incrémenté (clé primaire)

\* Un identifiant produit

\* Un identifiant visuel

\* Un drapeau signalant que ce visuel est un visuel principal pour ce produit

* Créer une table ‘suppliers’ (fournisseurs) comprenant les champs suivants :

\* Un index auto-incrémenté (clé primaire)

\* Un nom

\* Une adresse

* Créer un fichier .htaccess de façon à pouvoir accéder à l’API avec des URLs compatibles SEO.

RewriteEngine On # Activer le module Rewrite

RewriteRule ^produits/?$ produits.php **[**NC,L**]**

RewriteRule ^**[**^/**]**+/**(**\d+**)**$ produits.php?id=**$1**

* Créer un fichier permettant la connexion à la base de données.
* Créer un fichier en y et incluant le fichier de connexion à la base pour accéder aux données.

Dans ce fichier, on utilise l’entrée du tableau $\_SERVER[‘REQUEST\_METHOD’] pour accéder aux informations sur l’action demandée.

Au moyen d’une boucle switch/case sur l’action demandée (GET, POST, PUT, DELETE, etc.), coder les fonctions qui permettent de :

* Récupérer les données des produits (méthode GET) dont l’identifiant est fourni dans l’URL et tous les produits si aucun identifiant n’est transmis dans l’URL.

On envoie les informations des produits sous forme d’objet JSON au moyen de la fonction json\_encode.

* Ajouter les données d’un produit (méthode POST). Pour récupérer les valeurs transmises, utiliser les flux d’entrée :

$POST = array(); //tableau qui va contenir les données reçues

parse\_str(file\_get\_contents('php://input'), $POST);

Dans un premier temps, pour tester l’API en création, on pourra écrire le test unitaire suivant :

<?php

$url = "http://localhost/MDM/api/product"; // modifier le produit 1

$data = array(

    'code' => 'MOD',

    'description' => 'Moutarde de Dijon',

    'price' => 225,

    'category' => 1,

    'statut' => 2,

    'supplier' => 3,

    'purchase' => '2022-04\_01 10:40:00',

    'expire' => '2023-04-01 10:40:00'

);

$ch = curl\_init($url);

curl\_setopt($ch, CURLOPT\_RETURNTRANSFER, true);

curl\_setopt($ch, CURLOPT\_CUSTOMREQUEST, "POST");

curl\_setopt($ch, CURLOPT\_POSTFIELDS, http\_build\_query($data));

$response = curl\_exec($ch);

var\_dump($response);

if (!$response) {

    return false;

}

* Modifier un enregistrement dans la base au moyen d’une requête http PUT. Pour récupérer les valeurs transmises, utiliser les flux d’entrée :

**$\_PUT** = **array()**; //tableau qui va contenir les données reçues

parse\_str**(**file\_get\_contents**(**'php://input'**)**, **$\_PUT)**;

Pour tester la modification d’un produit en base, envoyer une requête PUT via cURL :

<?php

$url = "http://localhost/MDM/api/product/1"; // modifier le produit 1

$data = array(

    'code' => 'TVF',

    'description' => 'Thé vert saveur framboise en sachets',

    'price' => '149',

    'category' => '4',

    'status' => '2',

    'supplier' => '1',

    'purchase' => '2021-04\_01 10:40:00',

    'expire' => '2023-04-01 10:40:00'

);

$ch = curl\_init($url);

curl\_setopt($ch, CURLOPT\_RETURNTRANSFER, true);

curl\_setopt($ch, CURLOPT\_CUSTOMREQUEST, "PUT");

curl\_setopt($ch, CURLOPT\_POSTFIELDS, http\_build\_query($data));

$response = curl\_exec($ch);

var\_dump($response);

if (!$response) {

    return false;

}

?>

NB : si la catégorie est différente de celle déjà présente en base, alors on ajoute cette nouvelle catégorie dans la table de liaisons, sinon on ne fait rien.

* Supprimer un enregistrement dans la base de données en utilisant une requête http DELETE. L’identifiant du produit à supprimer est contenu dans l’URL. Prévoir la mise à jour des tables de liaisons des catégories et des visuels.

Pour tester la suppression via l’API, envoyer une requête DELETE via cURL.

**<**?php

**$url** = "http://127.0.0.1/api/produits/1"; // supprimer le produit 1

**$ch** = curl\_init**()**;

curl\_setopt**($ch,** CURLOPT\_URL, **$url)**;

curl\_setopt**($ch,** CURLOPT\_CUSTOMREQUEST, "DELETE"**)**;

curl\_setopt**($ch,** CURLOPT\_RETURNTRANSFER, **true)**;

**$response** = curl\_exec**($ch)**;

var\_dump**($response)**;

curl\_close**($ch)**;

?**>**

* Dupliquer un enregistrement de la base de données en utilisant une requête http personnalisée DUPLICATE. L’identifiant du produit à dupliquer est contenu dans l’URL. Prévoir la mise à jour des tables de liaisons des catégories et des visuels.

Pour tester la duplication via l’API, envoyer une requête DUPLICATE via cURL.

<?php

$url = "http://localhost/MDM/api/product/2"; // modifier le produit 1

$data = array(

    'code' => 'TVF2',

    'description' => 'Thé vert saveur framboise en sachets',

    'price' => '149',

    'category' => '4',

    'status' => '2',

    'supplier' => '1',

    'purchase' => '2021-04\_01 10:40:00',

    'expire' => '2023-04-01 10:40:00'

);

$ch = curl\_init($url);

curl\_setopt($ch, CURLOPT\_RETURNTRANSFER, true);

curl\_setopt($ch, CURLOPT\_CUSTOMREQUEST, "DUPLICATE");

curl\_setopt($ch, CURLOPT\_POSTFIELDS, http\_build\_query($data));

$response = curl\_exec($ch);

var\_dump($response);

if (!$response) {

    return false;

}

* Réaliser les mêmes tests en utilisant l’application Postman. https://www.postman.com/
* Faire quelques captures d’écran depuis Postman pour github.

1. Exploiter l’API

* Maquetter et réaliser une UI pour afficher :

\* Les produits et leur visuel principal en taille réduite (64 x 64 par ex.) sous forme de tableau :

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Visuel principal | Description | Prix en € | Date de péremption | Catégorie(s) | Statut | Suppression/ modification statut |

Un survol de la souris sur le visuel permet l’affichage du visuel en taille réelle.

\* Les mettre à jour. La seule mise à jour autorisée via l’UI est la modification du statut. Utiliser un select/option côté front. Cette modification est implémentée par appel AJAX.

\* Les supprimer. Une fenêtre popup valide le choix de l’utilisateur.

\* Il n’y a pas de création pour le moment.

* Prévoir des filtres permettant d’afficher les produits par catégorie, par statut ou par description en recherche fulltext.

1. Import / Export

La liste des produits est mise à jour régulièrement par le service achat qui émet à intervalles régulier un fichier Excel. Ce fichier est déposé par FTP dans le dossier input/. Le fichier est traité par l’application. On simulera l’opération qui est normalement automatique par un bouton ‘Importer’ sur l’UI. Optionnel : Une fois le fichier traité, il est déplacé dans un répertoire archives/.

Le traitement du fichier consiste en une opération de lecture du fichier Excel au moyen de la bibliothèque PHPSpreadSheet.

* Installer la bibliothèque PHPSpreadSheet via composer depuis le dossier du projet

composer require phpoffice/phpspreadsheet

* Tester la bibliothèque avec ce fichier (attention à bien vérifier le chemin vers autoload.php) :

**<?php**

**require** 'vendor/autoload.php';

**use** **PhpOffice**\**PhpSpreadsheet**\**Spreadsheet**;

**use** **PhpOffice**\**PhpSpreadsheet**\**Writer**\**Xlsx**;

$spreadsheet = **new** Spreadsheet();

$sheet = $spreadsheet->getActiveSheet();

$sheet->setCellValue('A1', 'Hello World !');

$writer = **new** Xlsx($spreadsheet);

$writer->save('hello world.xlsx');

Après exécution, un fichier hello world doit se situer dans le répertoire du fichier de test ci-dessus.

* Réaliser un script d’import en base de données du fichier XLS joint au CDC.

Outre l’import des données en base, le script réalise les actions suivantes :

1. Créer un fichier Excel par catégories de produits listant les nouveaux produits (création en base) et les produits dont le statut est passé de ‘En cours d’approvisionnement’ à ‘En stock’ (update), Ces fichiers ont pour vocation à être communiqués aux employés afin que les produits soient effectivement mis en rayon.
2. Mettre à jour la table ‘visuels’ en cas d’apparition d’un nouveau visuel dans le fichier d’import XLS
3. Mettre à jour la table de liaisons produits/visuels. La liaison avec le produit s’effectue grâce au nom du fichier image qui inclue le code produit. Si le fichier n’existe pas dans le dossier ‘assets/’, un visuel ‘Visuel-non-disponible.jpg’ sera lié au produit. De même, si le nom du visuel principal est manquant dans le fichier d’import, un visuel ‘Visuel-non-disponible.jpg’ sera lié au produit.

Pour que les mises à jour des statuts des produits soient prises en compte par le service achat, il est nécessaire de réaliser un export de la base de données sous la forme d’un fichier Excel utilisant la même trame que le fichier d’import.

On simulera cette opération en principe automatique à l’aide d’un bouton ‘Exporter’ qui déclenchera quatre opérations :

1. La création d’un fichier Excel d’export de la BDD à date, déposé dans un répertoire ‘export/’ sous le nom ‘export\_YYYY\_MM\_JJ\_HH\_MM\_SS.xls’
2. La création des bons de commande des produits dont le statut est ‘épuisé’ au format PDF, déposés dans un répertoire ‘commandes’ sous le nom ‘cmd\_nom-du-produit\_YYYY\_MM\_JJ.pdf’
3. La mise à jour du statut des produits de ‘Epuisé’ à ‘En cours d’approvisionnement’
4. Tous les produits dont la date de péremption est égale à la date du jour passent du statut ‘En stock’ à ‘Retiré des rayons’
5. Créer un fichier Excel listant les produits ‘Retiré des rayons’ qui sera à communiquer aux employés afin que les produits soient effectivement retirés.